Next Generation Workload Mangement System for Big Data

Kaushik De

Univ. of Texas at Arlington

BNL HPC Workshop

June 4, 2013

Big Data

- What is Big Data?
  - Many definitions – much hype
  - Common denominator – large data volumes
  - Huge variation in how Big Data applies to different scientific disciplines
- Focus of this talk
  - Specific Big Data use case in science == large data volume in highly distributed heterogeneous computing environment for collaborative science

Why this Use Case?

- Large data volume in science
  - Growth in storage and processing power has enabled scientific studies with huge data volumes
  - Almost every scientific discipline has benefited
- Collaborative science
  - Scientific problems impossible for small groups to solve can now be tackled by large collaborations
- Distributed computing infrastructure
  - Can grow/shrink to meet changing needs
  - But often simply to pool resources

Why WMS for Big Data?

- Workload Management System (WMS)
  - Most Big Data paradigms assume individual researcher (or small group) using large data store
  - Collaborative science requires new model
  - Complex data usage models
    - Multiple custom applications to be supported
    - Data is distributed and accessed worldwide
    - Large number of users requiring quick turn-around
    - Multi-step work flows/data processing
    - Flexible breakdown of processing ...
  - Each requirement leads to a different specialized product – WMS provides integrated solution

A Successful Example

- PanDA
  - Production and Distributed Analysis system developed for the ATLAS experiment at the LHC
  - Deployed on WLCG resources
  - Now also used by AMS and CMS experiments
  - Common Analysis Framework project with CERN IT
  - Large data volume – hundreds of petabytes
  - Distributed resources – hundreds of computing centers worldwide
  - Collaborative – thousands of scientific users
  - Complex work flows, multiple applications, flexible processing chunks, fast turn-around ...

References

- https://twiki.cern.ch/twiki/bin/viewauth/Atlas/PanDA
- http://www.usatlas.bnl.gov/twiki/bin/view/PanDA/WebHome
- http://panda.cern.ch:25880/server/pandamon/query
- Recent Improvements in the ATLAS PanDA Pilot, P. Nilsson, CHEP 2012, United States, May 2012
- PD2P : PanDA Dynamic Data Placement for ATLAS, T. Maeno, CHEP 2012, United States, May 2012
- Evolution of the ATLAS PanDA Production and Distributed Analysis System, T. Maeno, CHEP 2012, United States, May 2012

A bit of History

- The ATLAS experiment at the LHC
  - Well known for recent Higgs discovery
  - Search for new physics continues
- PanDA project was started in Fall 2005
  - Goal: An automated yet flexible workload management system which can optimally make distributed resources accessible to all users
  - Originally developed for US physicists
- Adopted as the ATLAS wide WMS in 2008 (before first LHC data in 2009)
- In use for all ATLAS computing applications

Why not use Traditional HPC/HTC Systems?

- HPC and HTC has worked well for science in the past using sophisticated batch systems
  - Individual researchers get allocations
  - Executables are built on local architecture
  - Jobs run (scheduled through batch systems)
  - New features (grid access, remote IO, network reservations...) still geared to individual researcher
  - Originally not designed for large data volumes
- LHC experiments chose distributed WLCG model of resources for initial use
- Now turning to HPC/HTC systems through PanDA

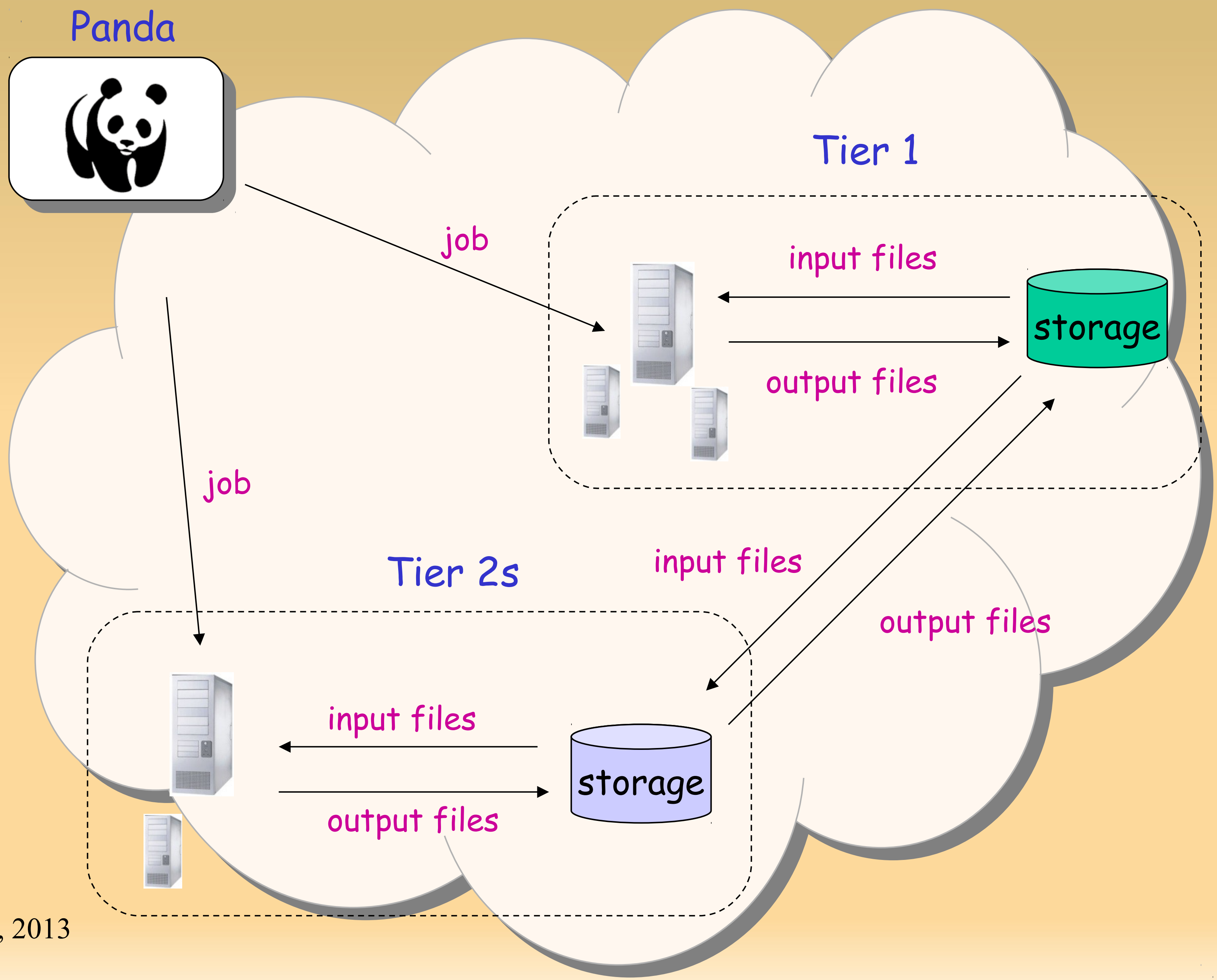
PanDA Philosophy

- PanDA WMS design goals:
  - Achieve high level of automation to reduce operational effort for large collaboration
  - Flexibility in adapting to evolving hardware and network configurations over many decades
  - Support diverse and changing middleware
  - Insulate user from hardware, middleware, and all other complexities of the underlying system
  - Unified system for production and user analysis
  - Incremental and adaptive software development

PanDA Basics

- Key features of PanDA
  - Pilot based job execution system
    - ATLAS work is sent only after execution begins on CE
    - Minimize latency, reduce error rates
  - Central job queue
    - Unified treatment of distributed resources
    - SQL DB keeps state - critical component
  - Automatic error handling and recovery
  - Extensive monitoring
  - Modular design

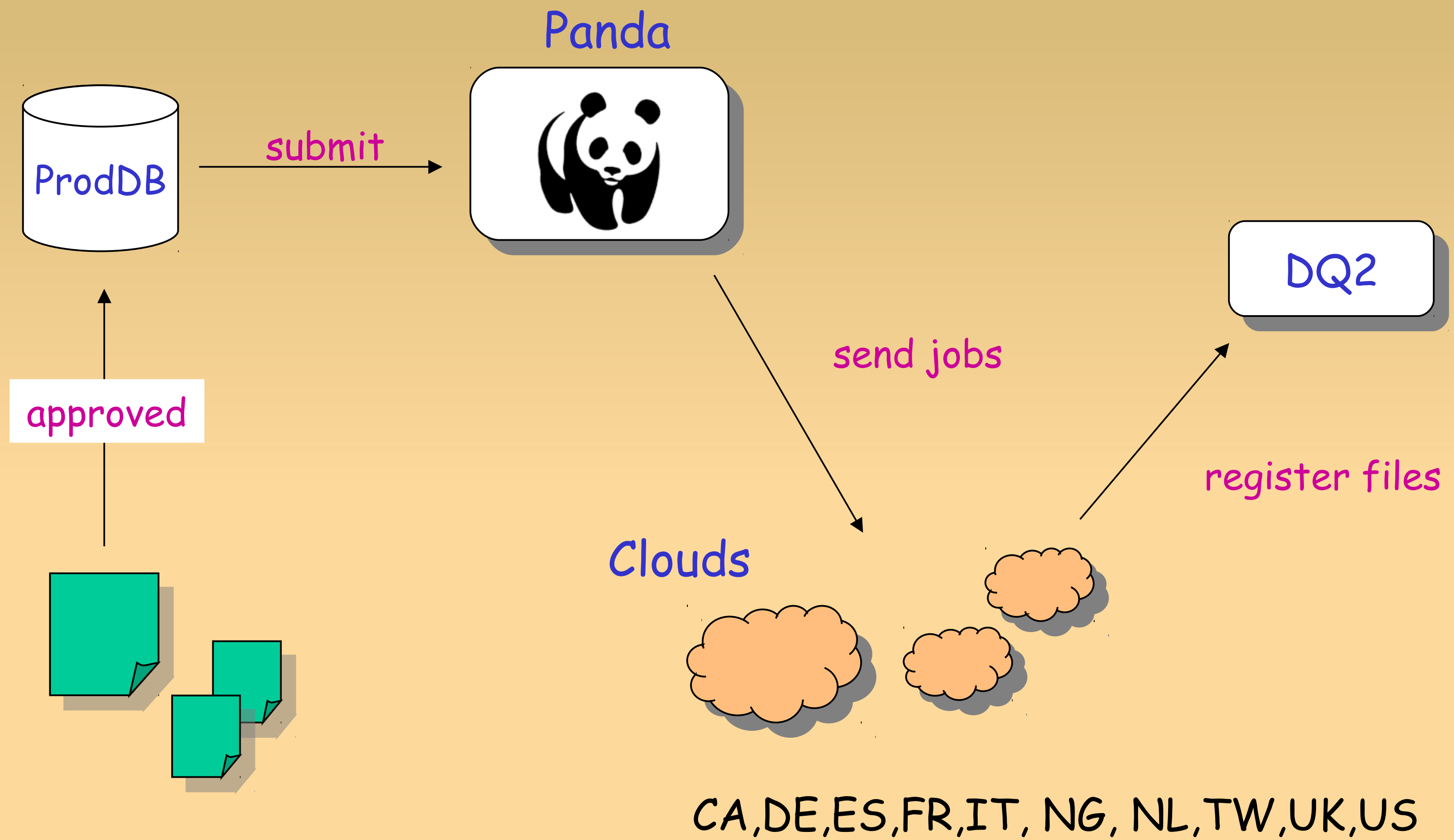
Simplified View



PanDA Components

- PanDA server
- Database back-end
- PanDA pilot system
  - Job wrapper
  - Pilot factory
- Brokerage
- Dispatcher
- Information system
- Monitoring systems

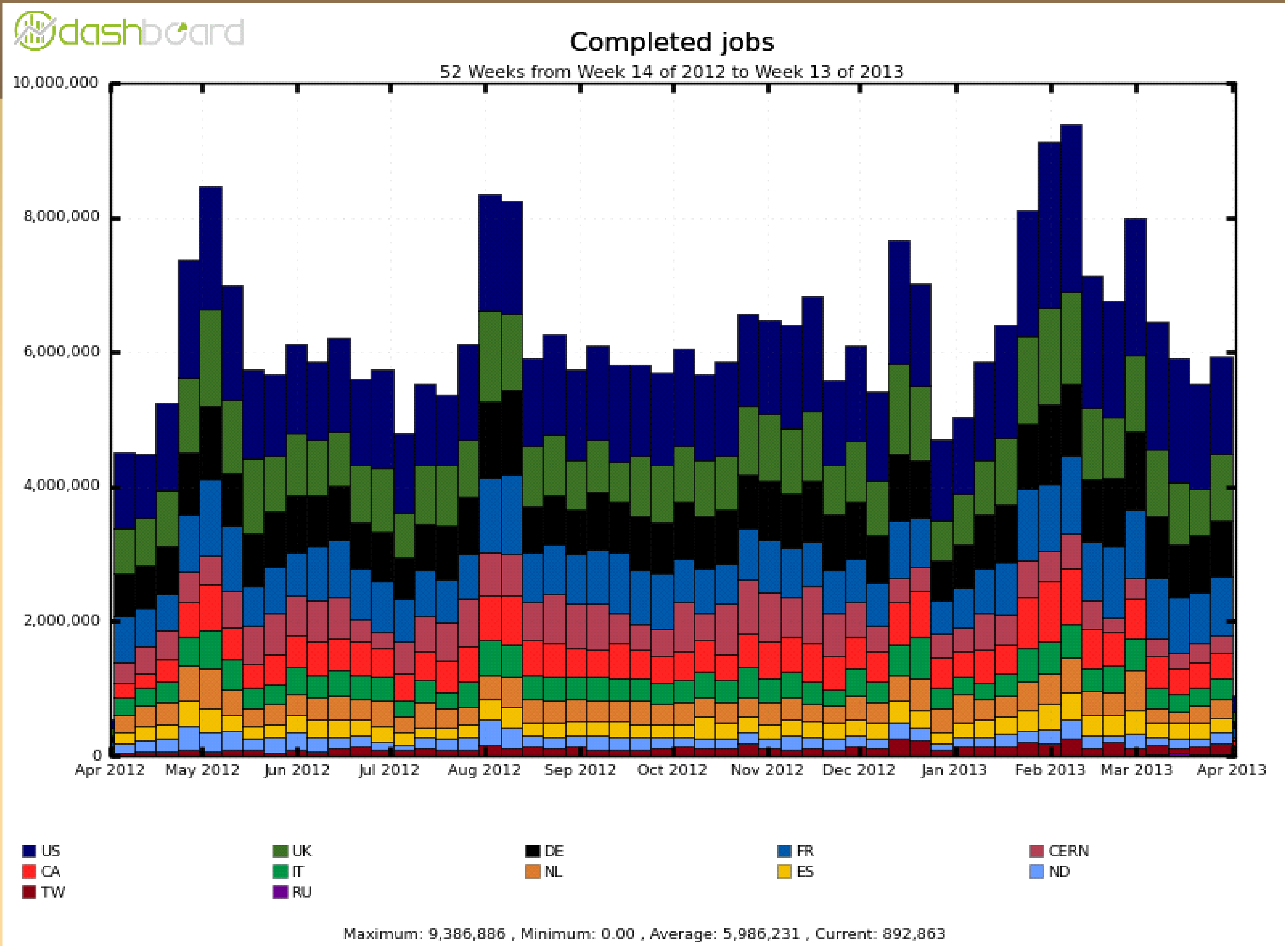
PanDA Design



- HTTP/S RESTful communication (curl+grid proxy+python)
- GSI authentication via mod_gridsite
- Workflow is maximally asynchronous

What is a Job

- Basic unit of work is a job:
  - Executed on a CPU resource/slot
  - May have inputs
  - Produces outputs
- ProdSys – layer above PanDA to create jobs from ATLAS physics 'tasks'
- User analysis work divided into jobs by PanDA
- Pilot may run multiple jobs on request
- Current scale – one million jobs per day

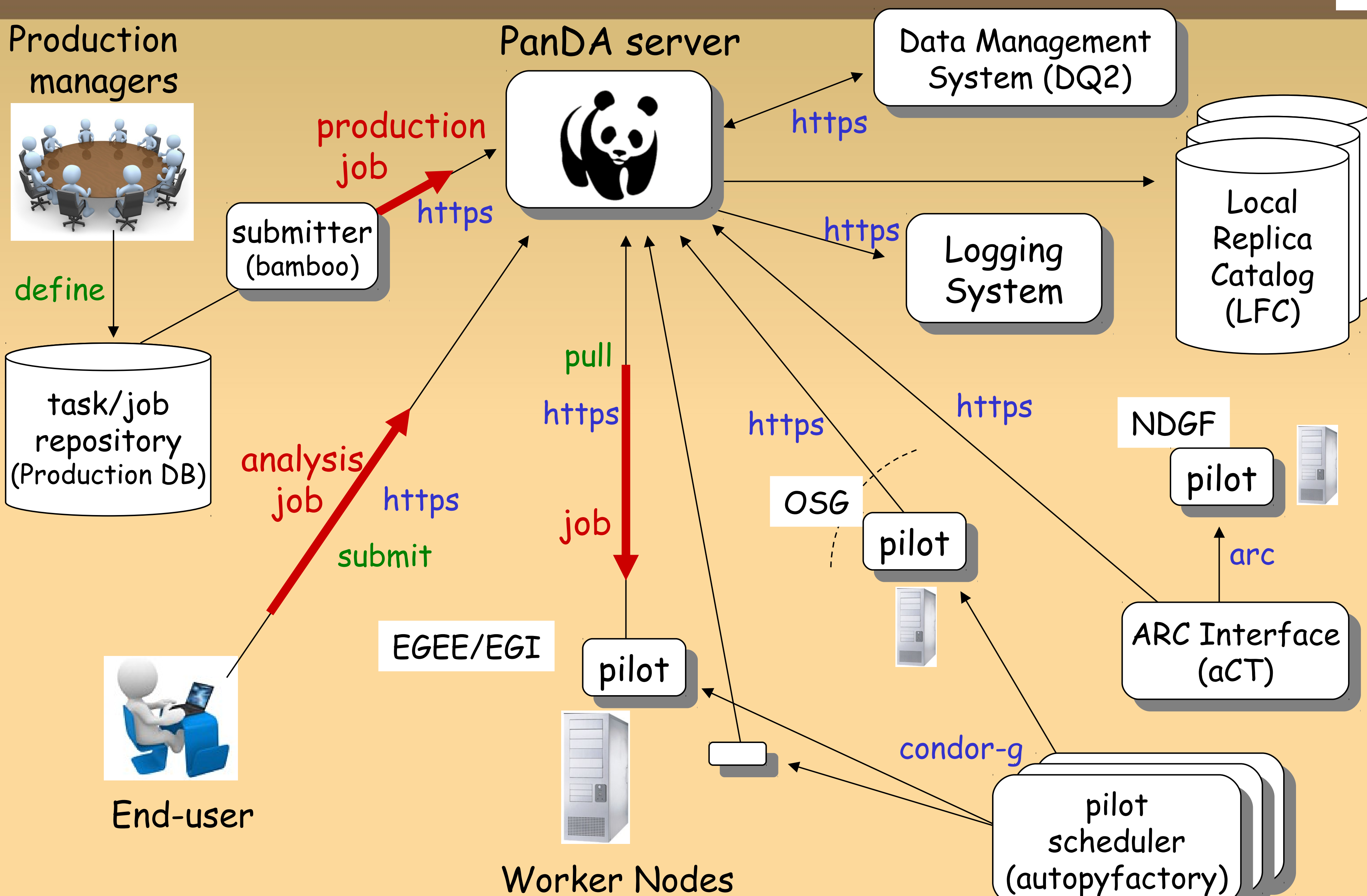


dashboard
Completed jobs
52 Weeks from Week 14 of 2012 to Week 13 of 2013
10,000,000
8,000,000
6,000,000
4,000,000
2,000,000
0
Apr 2012
May 2012
Jun 2012
Jul 2012
Aug 2012
Sep 2012
Oct 2012
Nov 2012
Dec 2012
Jan 2013
Feb 2013
Mar 2013
Apr 2013
US
UK
DE
FR
CERN
CA
IT
NL
ES
ND
TW
RU
Maximum: 9,386,886 , Minimum: 0.00 , Average: 5,986,231 , Current: 892,863

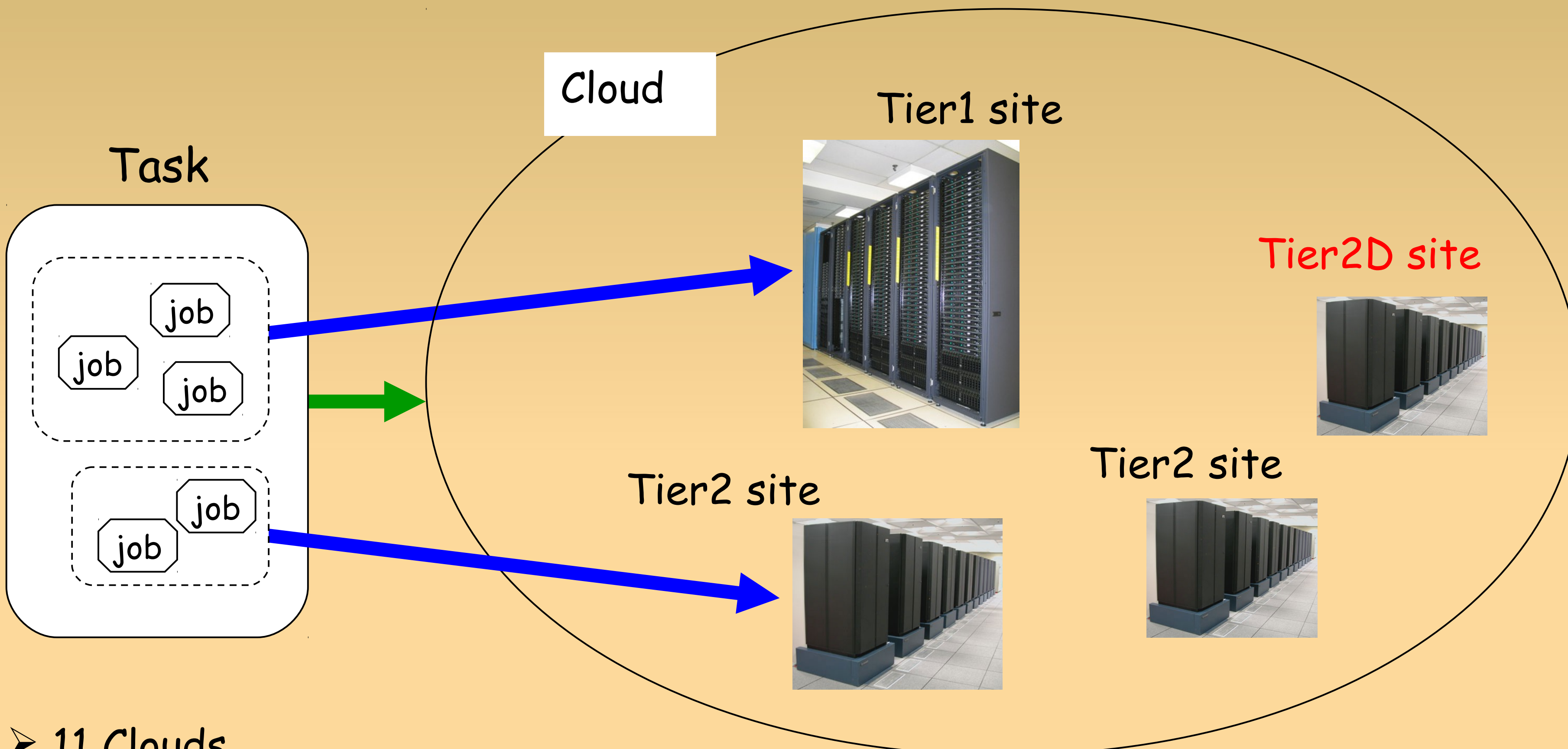
What is a Pilot Job

- Lightweight execution environment to prepare CE, request actual payload, execute payload, and clean up
- Handles data stage-in and stage-out between worker node disk and local SE
- Pilot jobs started by Job Scheduler(s); actual ATLAS job (payload) is scheduled when CPU becomes available, leading to low latency
- Monitoring thread, job recovery, experiment specific setup and post processing...

Workload Management



ATLAS Computing Model



- 11 Clouds
  10 T1s + 1 T0 (CERN)
  Cloud = T1 + T2s + T2Ds (except CERN)
  T2D = multi-cloud T2 sites
- 2-16 T2s in each Cloud

Task → Cloud
Task brokerage
Jobs → Sites
Job brokerage

Task Brokerage

- Matchmaking per cloud is based on:
  - Free disk space in T1 SE, MoU share of T1
  - Availability of input dataset (a set of files)
  - The amount of CPU resources = the number of running jobs in the cloud (static information system is not used)
  - Downtime at T1
  - Already queued tasks with equal or higher priorities
  - High priority task can jump over low priority tasks

Job Brokerage

- Brokerage policies define job assignment to sites
  - IO intensive or TAPE read -> T1
  - CPU intensive -> T1+T2s
  - Flexible: clouds may allow IO heavy jobs at T2s with low weight
- Matchmaking per site in a cloud
  - Software availability
  - Free disk space in SE, Scratch disk size on Worker Node (WN), Memory size on WN
  - Occupancy = the number of running jobs / the number of queued jobs, and downtime
  - Locality (cache hits) of input files

Job Dispatcher

- High performance/high throughput module
- Send matching job to CE upon pilot request
  - REST non-blocking communication
  - Different from brokerage, which is asynchronous
- Matching of jobs based on
  - Data locality
  - Memory and disk space
- Highest priority job is dispatched

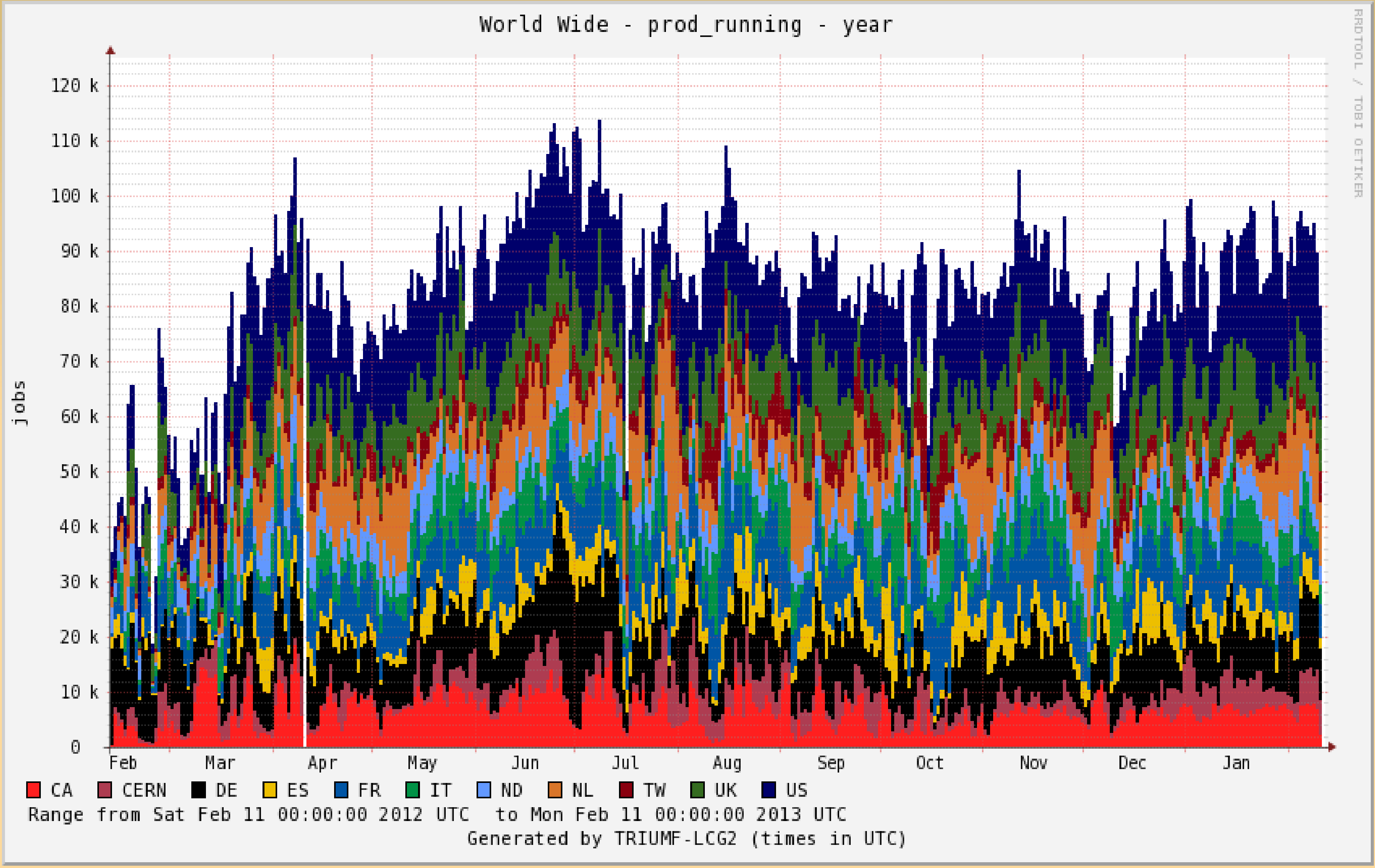
Data Management

- LFC file catalog
- Asynchronous file transfers by ATLAS DDM (DQ2)
  - Dispatch of input files from T1 SE to T2 SE, pre-staging of input files on T1 TAPE SE, aggregation of output files to T1 SE from T2 SE
  - CE CPU is not wasted waiting for transfers – pilot job starts only after input files ready, and ends after output is put on local SE
- Reusing input/output files as caches
  - Cache lifetime defined per cloud, job brokerage takes cache hits into account
- HTTPS message exchange between PanDA and DDM

Example of Flexibility

- PanDA supports multiple DDM solutions
  - Caching without LFC lookup
  - Pandamover file transfer (using chained Panda jobs)
  - Direct access if requested (by task or site)
  - Customizable lsm (local site mover)
  - Multiple default site movers are available
  - Flexible dataset sizing/containers for scalability

Performance - Production

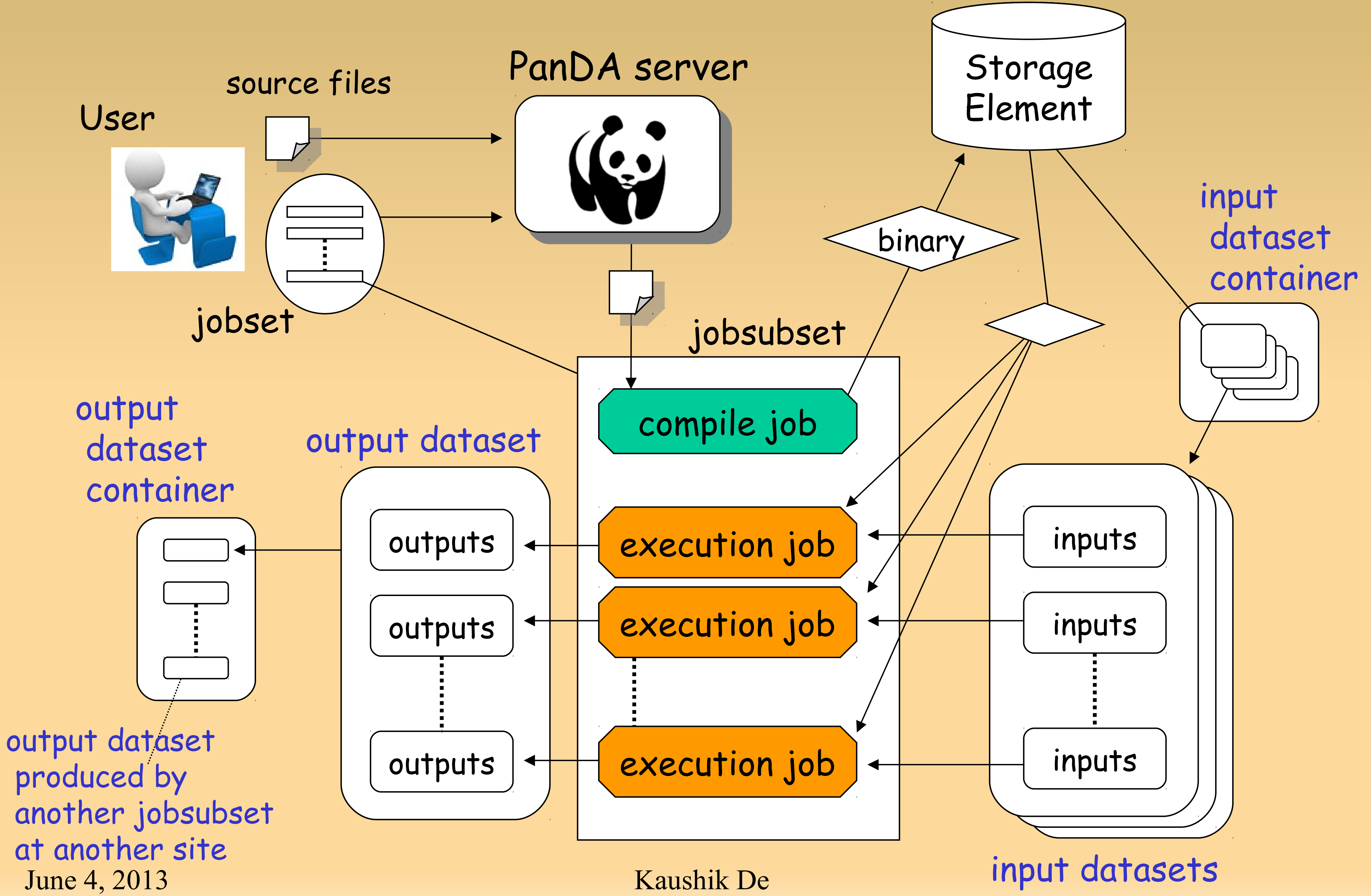


Average number of concurrently running jobs per day

User Analysis in PanDA

- Flexibility in job definition
  - Customization of source files: adding new algorithms to ATLAS application (athena) or arbitrary executables
- Fast turnaround for iteration
  - The user submits a user task (jobset) that is converted to many jobs for parallel execution, using pilot system for high throughput
- Jobs go to data
  - No input file dispatch, no output file aggregation from multiple jobs
  - Data Transfer Request Interface or Dynamic Data Placement: supports IO intensive workflows
- Dataset container (a set of datasets) as input and output
- Priority and quota system based on user or working group
- Unique users in the last 3 days : 407; 7: 556; 30: 870; 90: 1209

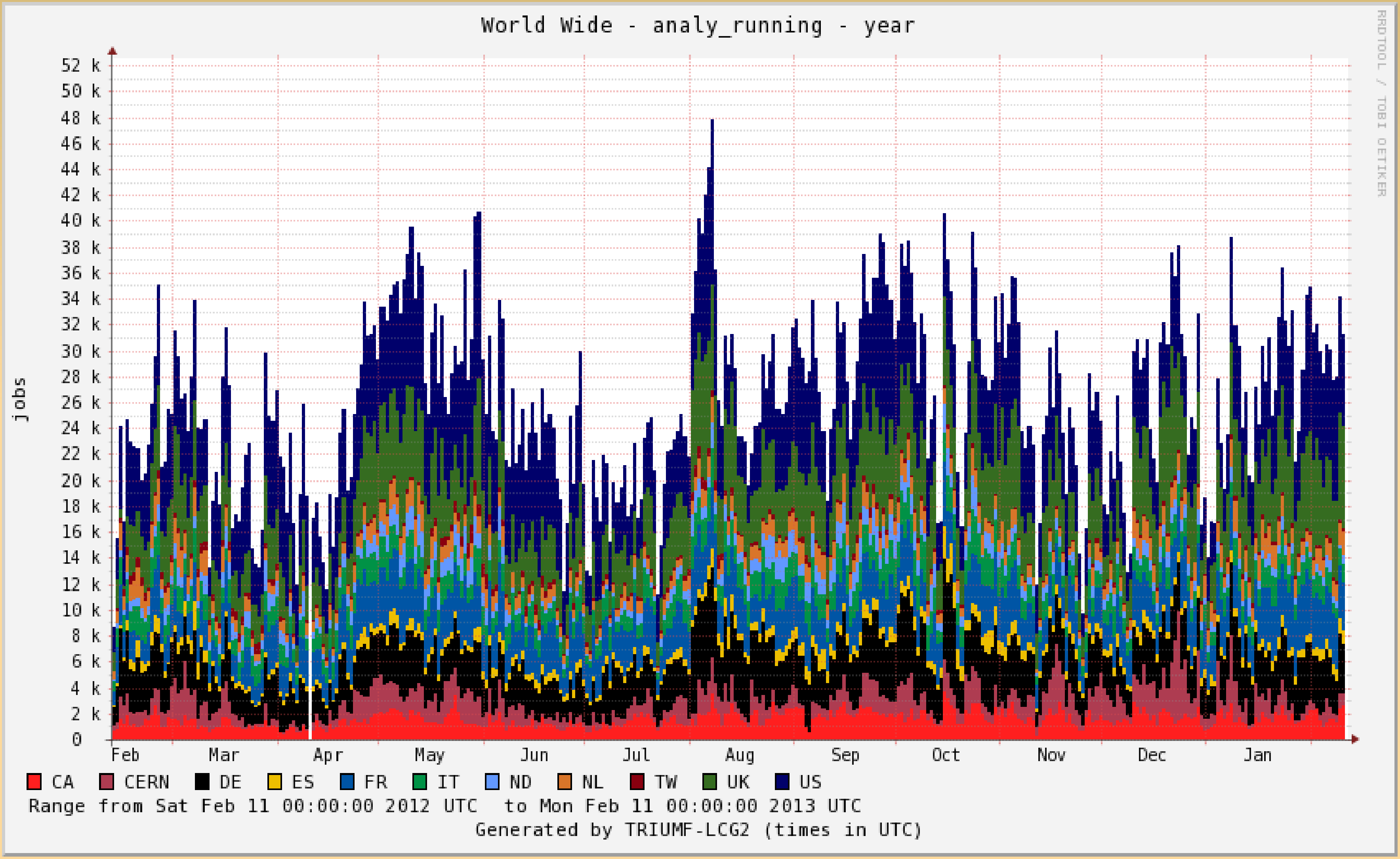
User Analysis Work Flow



Analysis Brokerage

- Works with jobsubset
  - A jobset may be split to be brokered to multiple sites
- Matchmaking per site without cloud-boundaries
  - Scratch disk size on WN, Memory size on WN
  - Software availability, Downtime
  - Occupancy = the number of running jobs / the number of queued jobs
  - Availability of input datasets

Analysis Performance



Average number of concurrently running jobs per day

PD2P – Recent Development

- PD2P = PanDA Dynamic Data Placement
- PD2P used to distribute data for user analysis
  - For production PanDA schedules all data flows, but initial computing model for user analysis was static distribution – PanDA sent jobs to data
  - Soon after LHC data started, we implemented PD2P
- Asynchronous usage based data placement
  - Repeated use of data → additional copies
  - Backlog in processing → additional copies
  - Rebrokerage of queued jobs use new data location
  - Deletion service removes less used data

Cloud Computing and PanDA

- ATLAS cloud computing group set up few years ago to exploit virtualization and clouds
  - PanDA queues in clouds – additional resources
  - Tier 3 in clouds – good for small institutes
- Excellent progress so far
  - Commercial clouds – invaluable Google, EC2
  - Helix Nebula for MC production (CloudSigma, T-Systems and ATOS – all used)
  - Futuregrid (U Chicago), Synnefo cloud (U Vic)
- Personal PanDA analysis queues being set up

Many Other Evolutions

- Federated storage (FAX)
  - Step by step plan to integrate into PanDA
  - First steps already successful
  - Different than current data management model
- JEDI – dynamic job definition
  - Higher level service to automatically define jobs from physics tasks
  - New level of brokerage
  - Better resource matching – especially MP jobs
- ……

Next Generation of WMS

- As PanDA usage grows beyond ATLAS:
  - Need common development environment
  - Separate core parts from experiment specific layers
  - Need packaging and regular releases
  - Modular – plug-in structure for most components
  - Availability of default plug-in's
- Support both small and large user bases
- Enhance and grow PanDA capabilities
  - Access to HPC resources, cloud resources
  - Integration with networking

DoE ASCR Project

- "Next Generation Workload Management and Analysis System for Big Data"
- 3 year DoE ASCR funding
  - Lead Institution: Brookhaven National Laboratory
  - Lead PI: Alexei Klimentov
  - Principal Investigators:
    - Brookhaven National Laboratory: Alexei Klimentov, Sergei Panitkin, Torre Wenaus, Dantong Yu
    - Argonne National Laboratory: Alexandre Vaniachine
    - The University of Texas at Arlington: Kaushik De, Gergely Zaruba

BigPanDA Work Packages

- WP1 (Factorizing the core): Factorizing the core components of PanDA to enable adoption by a wide range of exascale scientific communities (UTA, K.De)
- WP2 (Extending the scope): Evolving PanDA to support extreme scale computing clouds and Leadership Computing Facilities (BNL, S.Panitkin)
- WP3 (Leveraging intelligent networks): Integrating network services and real-time data access to the PanDA workflow (BNL, D.Yu)
- WP4 (Usability and monitoring): Real time monitoring and visualization package for PanDA (BNL, T.Wenaus)

BigPanDA Development Team

- Core ATLAS PanDA Team:
  - Tadashi Maeno, Paul Nilsson, many others
- New ASCR Hires:
  - BNL: Jaroslava Schovancova
  - UTA: Danila Oleynik, Artem Petrosyan, Mikhail Titov
- Integrated with many international teams:
  - CERN IT – CAF, CMS, AMS team
  - NorduGrid, Dubna and Kurchatov teams
- This meeting to discuss future directions